»Лекционен курс

»ООП1 (Java)

Въведение в ООП ›

Разработване на софтуер

Свойства на софтуера

- Няма изхабяване
  - при многократна употреба
- Лесен за копиране
  - също грешките
- Остарява
  - софтуерът постоянно се приспособява
- Дълго в употреба
- Трудно измерим
  - метрики: качество, количество
- Изключително комплексен
  - Реалният софтуер, използван в практиката е изключително комплексен
  - Принадлежи към едни от най-сложните артефакти, създавани от хората

Софтуер =
Програми, данни, документация

Качествени критерии за софтуерните продукти

- Коректност
- Стабилност: напр. при обработка на грешки
- Ефективност
- Удобен за потребителя
- Документираност:описание на програмите
- Модифицируемост
- Четаемост (разбираемост): коментари, смислен избор на означения, форматиране ...
- Многократна използваемост
- Модулност: декомпозиран на модули / компоненти
- Преносимост: върху различни компютърни конфигурации
- Интегрируемост: защита срещу неправомерен достъп
- ...

Разработване на софтуер

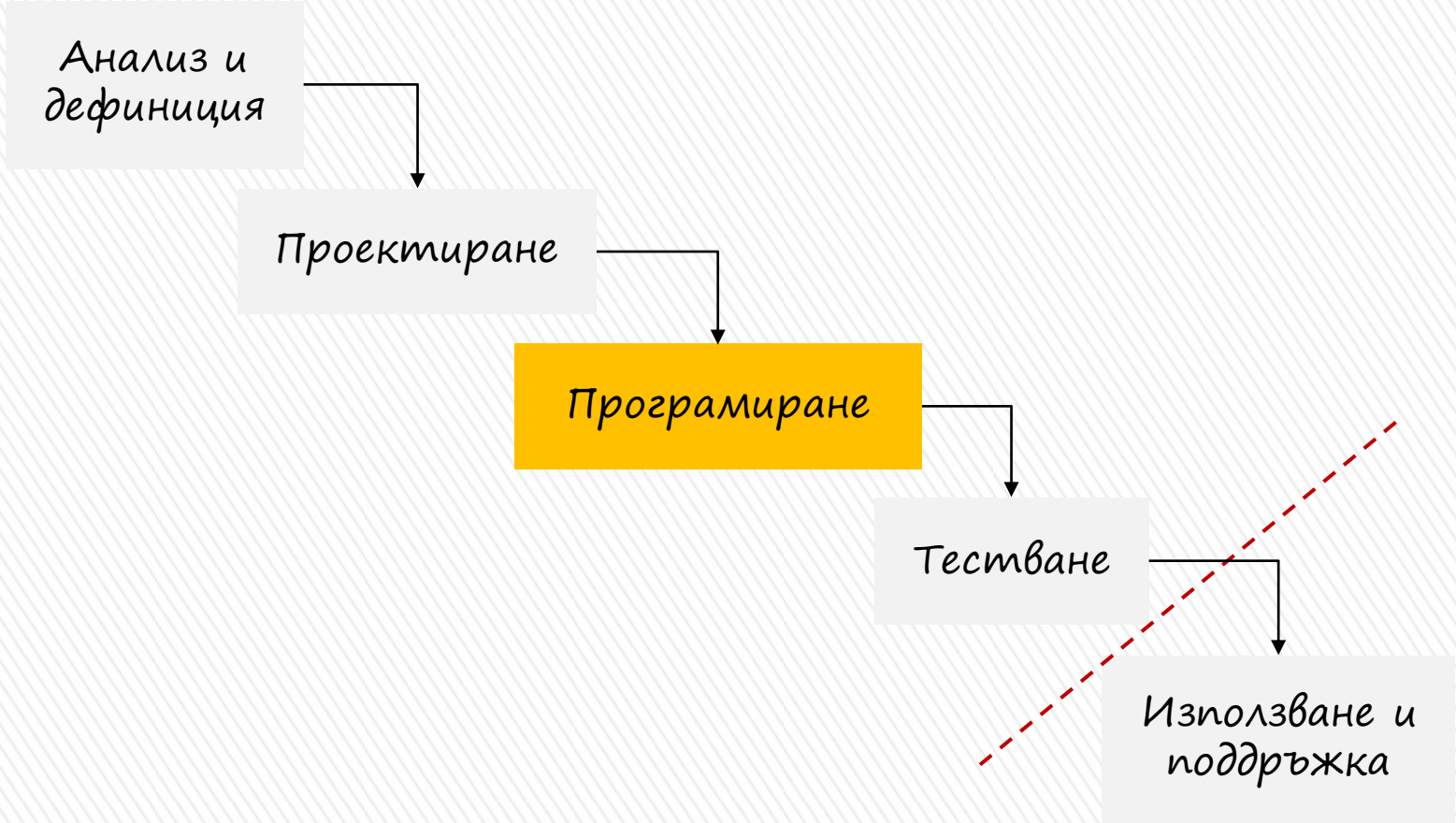
- След получаване на задача за разработване на софтуер:
  - Не трябва веднага да се захващаме с програмиране
  - Първо добре обмисляне на задачата, обсъждане на неясни постановки ...
- Програмирането е една малка част от цялостния софтуерен развоен процес
- Също така, особено съществени способности като:
  - Намиране на точна дефиниция на проблемите
  - Систематично търсене на грешки, ...
- Още при програмирането трябва да се мисли за доставката (поддръжката) на софтуера:
  - Четаеми, добре структурирани, добре коментирани програми
- Разработването на софтуер винаги предполага ‘работа в екип’

Модели: преглед

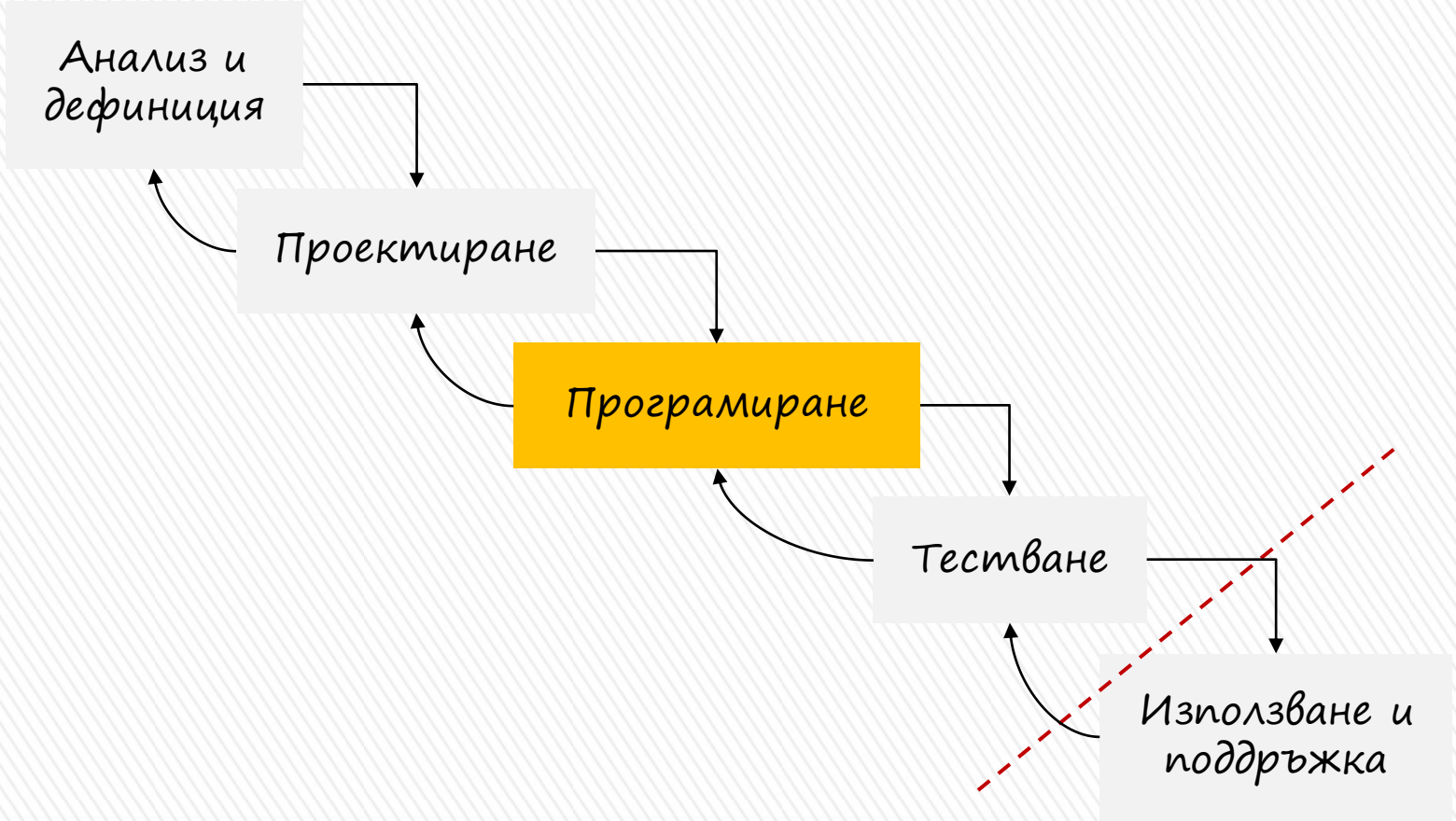
- Класически фазов модел (водопаден модел)
- Итеративен фазов модел (жизнен цикъл)
- Спирален модел
- Прототипиране (еволюционно разработване)
- Трансформационно разработване
- Използване на многократно използваеми компоненти

Лекционни курсове в областта "Софтуерно инженерство"

Пример: Класически водопаден модел

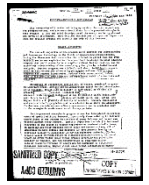


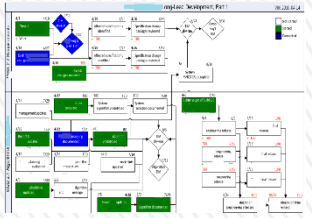
Пример: Итеративен фазов модел



Документи на развойния процес

Желания на клиента *Спецификация* *Проектиране* *Програмиране*





клиент анализатор проектант програмист

Студентско разработване на софтуер

Как студентите разработват софтуер?

Курсов проект

Чете проблема (ден 1) → Програмира (ден n-2) → Тества (ден n-1) → Документира (ден n-е) → Предава (ден n)

Сложност на софтуера

Сложност на софтуера

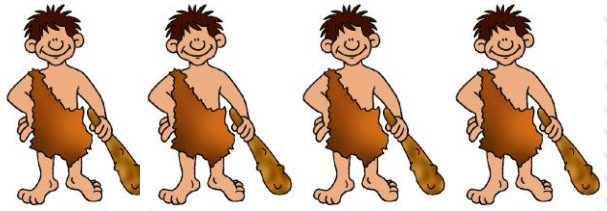
- Основен проблем на разработване на софтуер: комплексност
- ООП възниква за решаване на този проблем

Подходи - Декомпозиране

- Софтуерните системи не могат да бъдат напълно разбрани:
  - Както предварително при развоя
  - Така също и впоследствие, при използването им
- Първи принцип за овладяване комплексността на разработването на софтуер:
  - декомпозиция

Реална история … проблем

Разработчици

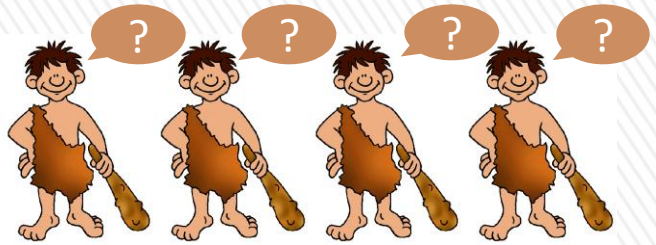


Никога не можем да го създадем … Не го разбираме …

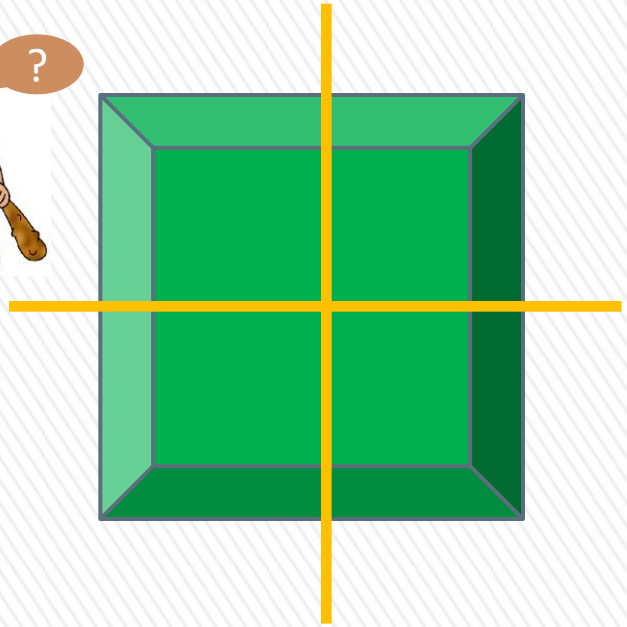
Софтуерът може да бъде много комплексен …

Реална история … решение

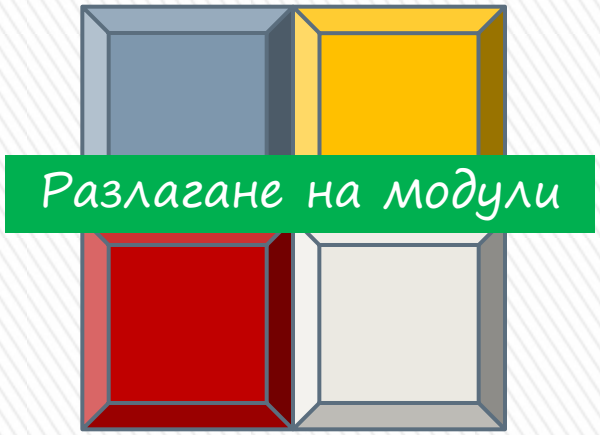
Разработчици



ДЕКОМПОЗИЦИЯ!!!

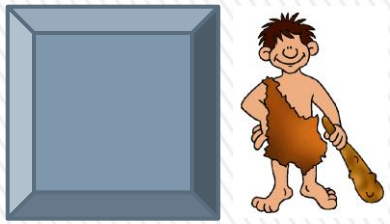


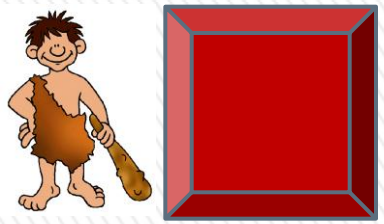
Речено … сторено



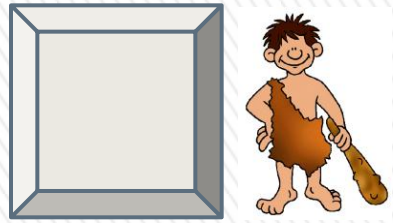
Овладяване на комплексността – разделяй и владей!

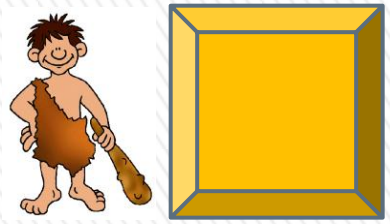
Една година по-късно …





Готови сме!
Интегрираме!





Една година по-късно …

Нещата не изглеждат така като сме ги замислили?

Овладяване на сложността

Овладяване на комплексността: декомпозиция

- Разлагане на софтуера на модули
- Всеки модул: поединично обхващаем
- Модули:
  - Разработване независимо от останалата част на системата
  - Семантично:
    - Модулите съответстват на подзадачи

Овладяване на комплексността: абстракция

- Софтуерните системи не могат да бъдат напълно разбрани:
  - Както предварително при развоя
  - Така също и впоследствие, при използването им
- Втори принцип за овладяване комплексността на разработването на софтуер:
  - абстракция

Овладяване на комплексността: абстракция

- Модулите са абстракции
- Останалата част от софтуерната система познава само:
  - Външното поведение на модулите
  - Не обаче детайли на реализацията – те трябва да бъдат скрити.

Интерфейс на модула

Една банална коледна история

Дядо Коледа е един весел старец, облечен в червен кожух, пристига на шейна, теглена от елени, ...

На Коледа през нощта Дядото остава подаръци под елхата.

От своя страна преди да легнат да спят децата му оставят шоколад и нещо топло за пиене (напр. мляко) ...

След Коледа (до следващата) Дядото се възстановява и подготвя списък с подаръци за следващата година ...

Една банална коледна история

Какво се случва след като децата легнат да спят на Коледа? ....

Дядо Коледа е един весел старец, облечен в червен кожух, пристига на шейна, теглена от елени, ...

На Коледа през нощта Дядото остава подаръци под елхата.

От своя страна преди да легнат да спят децата му оставят шоколад и нещо топло за пиене (напр. мляко) ...

След Коледа (до следващата) Дядото се възстановява и подготвя списък с подаръци за следващата година ...

Една банална коледна история

Какво се случва след като децата легнат да спят на Коледа? ....

Дядо Коледа е един весел старец, облечен в червен кожух, пристига на шейна, теглена от елени, ...

На Коледа през нощта Дядото остава подаръци под елхата.

От своя страна преди да легнат да спят децата му оставят шоколад и нещо топло за пиене (напр. мляко) ...

След Коледа (до следващата) Дядото се възстановява и подготвя списък с подаръци за следващата година ...

След като децата са заспали родителите им започват да изпълняват ролята на Дядо Коледа ....

Майката подрежда подаръците под елхата....

Бащата излива млякото обратно в бутилката (чашата трябва да бъде намерена празна, понеже ... Дядо Коледа е изпил млякото ...)...

Отваря шоколада и ... си налива чаша уиски (шоколадът трябва да го няма, понеже ... Дядо Коледа го е изял ...)...

Извод

Какъв е изводът от Коледната история?

Без декомпозиция, абстракция и скриване (капсулиране) ...

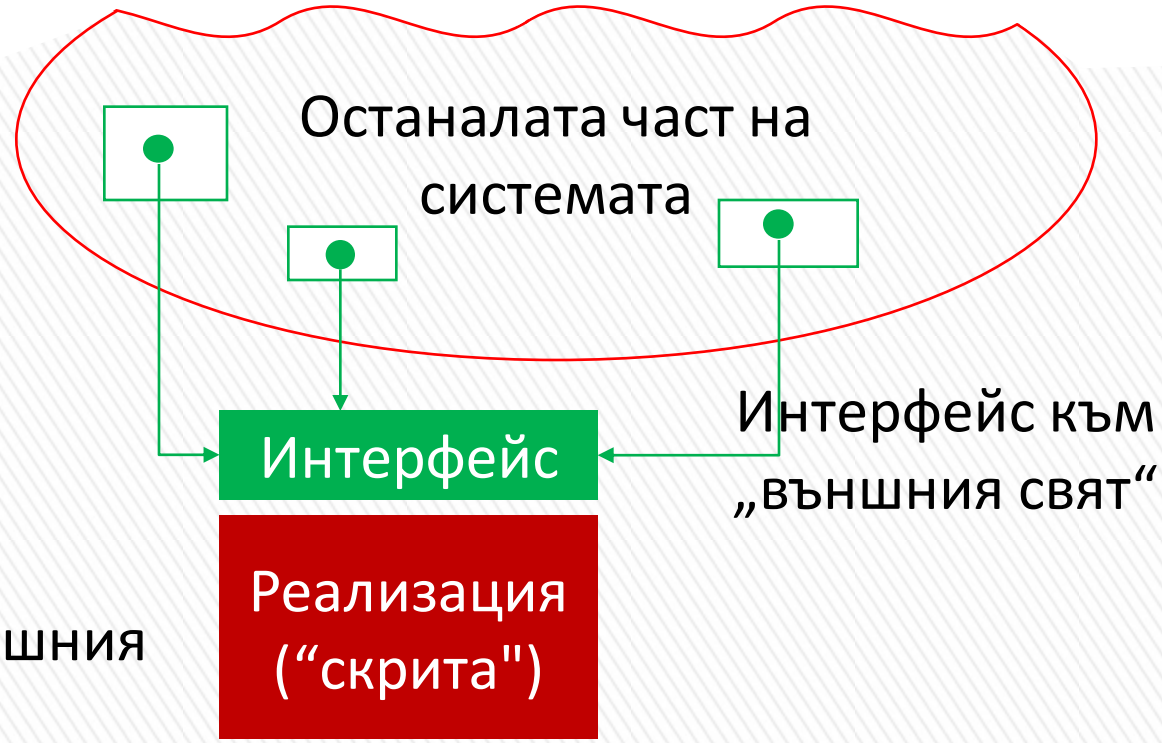
Коледата невъзможна !

Софтуерни абстракции

Основни характеристики на ООП

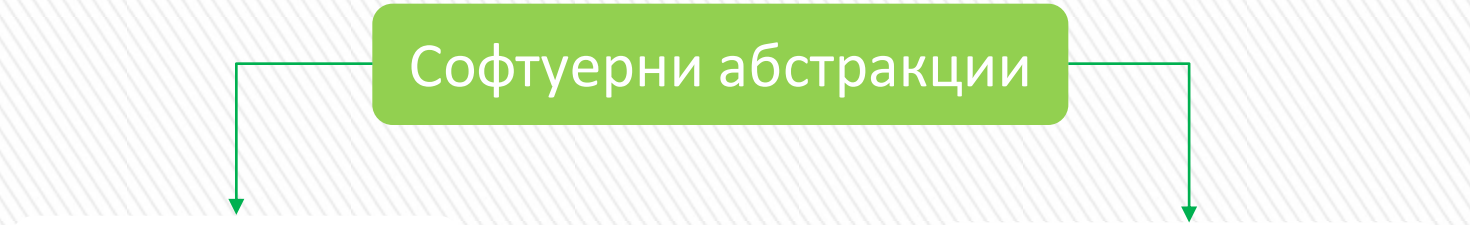
- Основни характеристики на ООП:
  - Капсулиране
  - Наследяване
    - + Обектите са основа, от която водят началото си други обекти
  - Полиморфизъм
  - Многократна използваемост
    - + Поведението на обектите е генетично: т.е. те могат да бъдат използвани в различни ситуации

Модули



В реализацията се абстрахираме от „външния свят“

Основни софтуерни абстракции



Процедурни

(алгоритмични)

Абстракции от алгоритъма

- ✓ Процедури
- ✓ Функции
- ✓ Методи

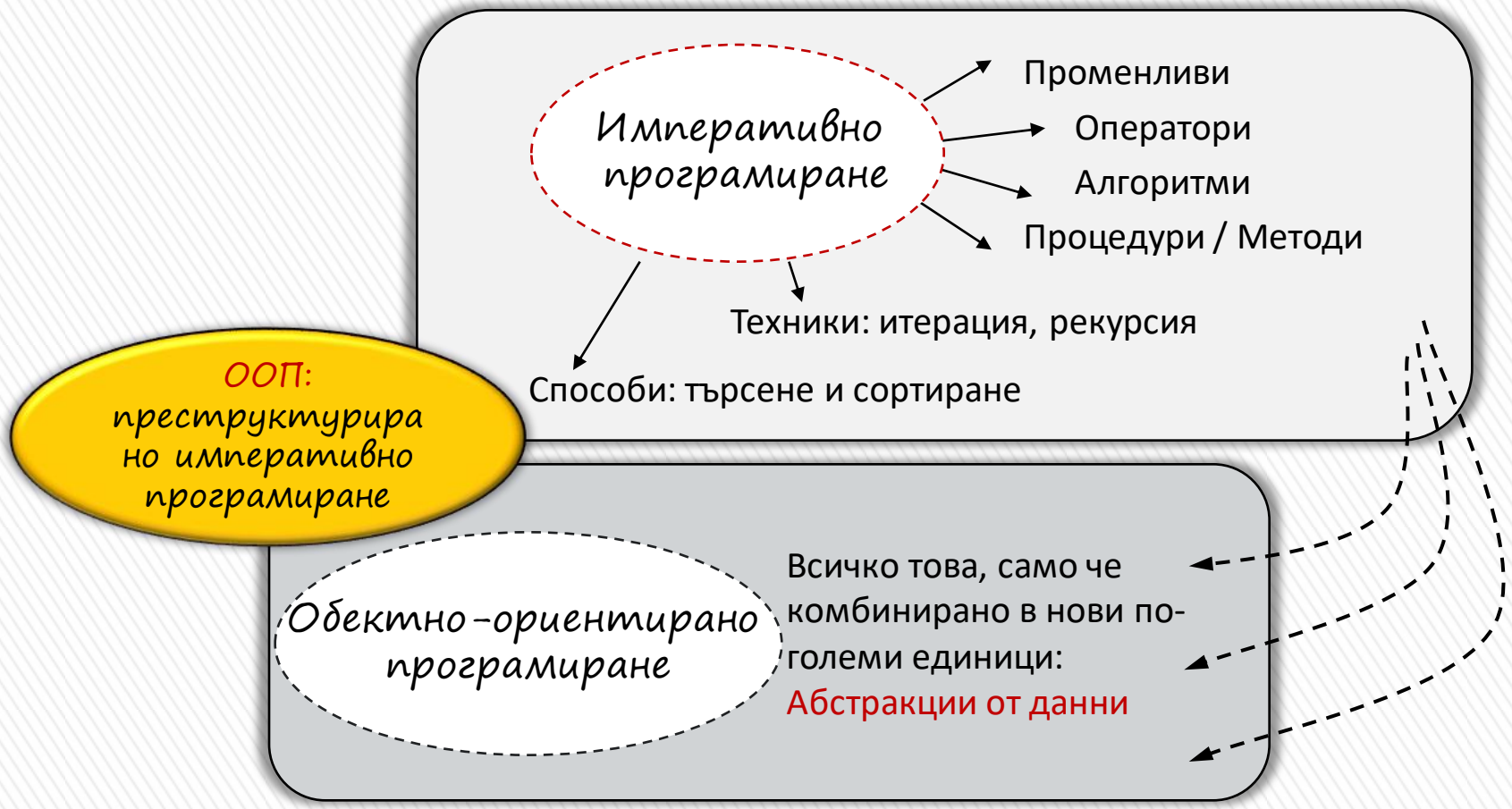
Данни

(абстрактни типове)

Абстракции от представяне на данни

- ✓ Модули (Modula-2)
- ✓ Класове (Java, C++)

Императивно програмиране: основно помощно средство на ООП



Абстрактни типове данни

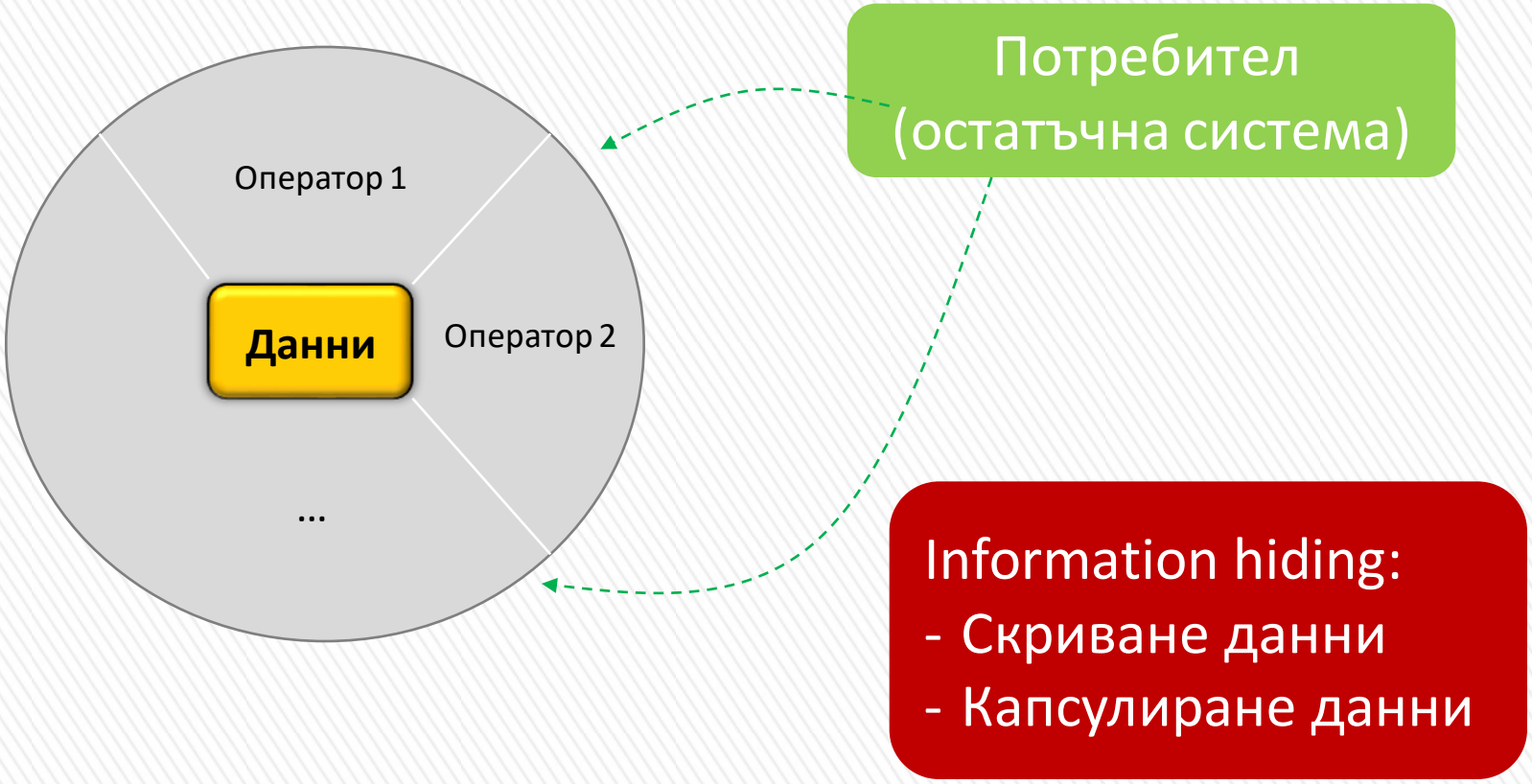
Единица от данни и операции

- **Операции:** служат за обработка на данни (инициализация, промяна, четене, изтриване)
- **Данни:** защитени/скрити от “външния свят"

Бележки:

- Основен принцип на разработването на ОО софтуер: „скриване на информация"
- Сравнение с императивното програмиране: данните и алгоритмите/операциите са разделени

Абстрактни типове данни:(information hiding)




Благодаря за вниманието!